

SAE AUTOMATISME

Introduction :

Dans ce projet l'objectif est d'automatiser une porte de garages (modèle réduit) de A à Z comprenant deux partie :

- Une partie électronique
- Une programmation sur automate Siemens

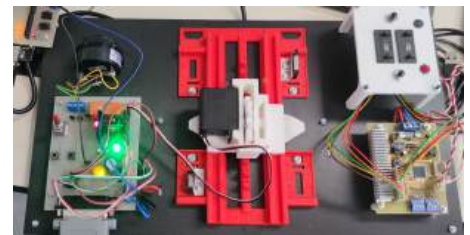
Dans ce projet nous utiliserons un automate de la marque Schneider Electric qui sera associé à une carte électronique (avec Arduino) :



Automate Schneider



+ Carte électronique



+ Maquette

Programmation en Grafset et en C pour la carte électronique .

Décomposition du projet :

-Carte électronique avec arduino :

- Fonction principale de la carte
- Raccordement de la carte d'interface Arduino à la maquette
- Programmation et envoie de la commande impulsionnelle , avec consigne manuelle

-Programmation M340

- Réalisation du raccordement entre l'automate (M340) et la maquette
- Raccordement entre M340 et l'interface Arduino
- Vérification du fonctionnement du servomoteur depuis la commande analogique automate
- Programmation en langage Grafcet

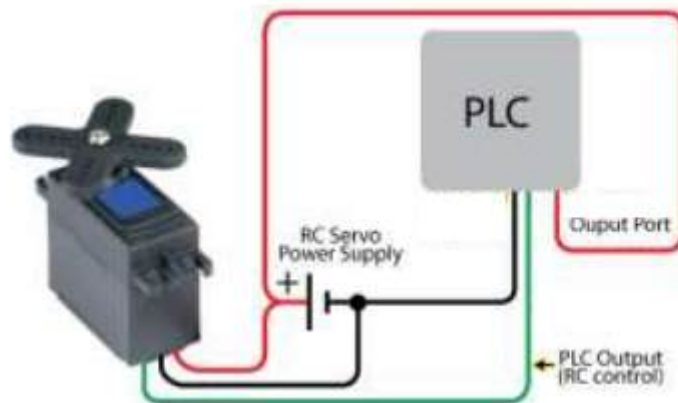
-Voies d'amélioration

- Réglage manuel des commandes limites de vitesse
- Prise en compte dans le programme automate

Carte électronique avec arduino :

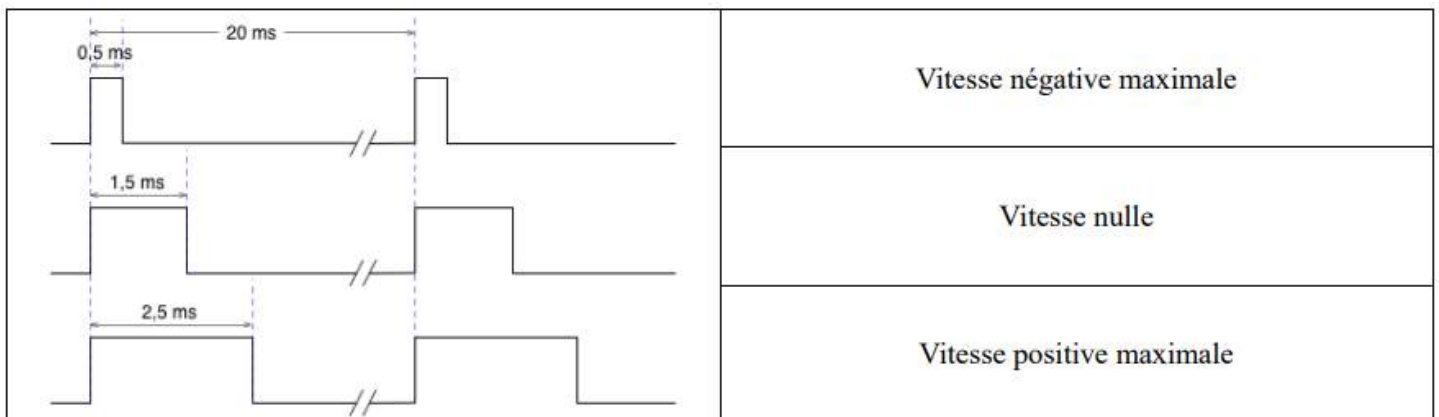
1. Fonction principale de la carte :

En effet, la maquette n'est pas réellement équipée d'un moteur à courant continu mais d'un servomoteur. Cet organe est un moteur équipé d'une électronique de contrôle qui gère la vitesse du moteur et son sens de rotation. Voici son schéma de câblage.



Il y a une masse (Noir), une alimentation (Rouge) et un signal de contrôle (Vert). La sortie MO_ON qui active le relais s'occupera de l'alimentation. La sortie SENS_MD affectera le signal de contrôle.

Le signal doit avoir la forme suivante :



Notre automate est censé pouvoir générer ces impulsions .On va donc se pencher sur les caractéristiques de notre matérielle.

Donner la valeur de la fréquence du signal de commande à délivrer par l'automate :

La valeur de fréquences du signal de commande à délivrer par l'automate est de 100Hz.

Rappeler les caractéristiques de sorties de l'automate Siemens S7-1214 DC/DC/DC (nombre et type (relais ou statique et fréquence de communication max)):

- Le nombre de sorties minimum est de 5 statique et 10 statique maximum.
- La fréquence de communication maximum est de 100KHz(à la sortie Qa.0 à Qa.3).

Donner les caractéristiques de la carte des sorties TOR de notre automate :

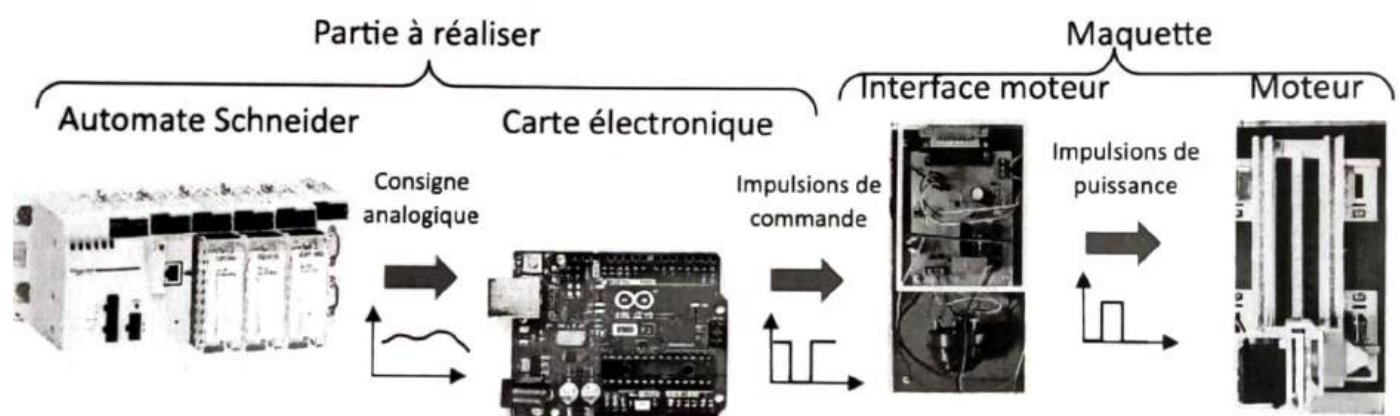
La carte de sortie TOR de notre automate:

- Il a 16 sortie Relays.
- IL a une fréquence maximum de communications de 1.2ms($\approx 833.33\text{Hz}$).

Conclure sur l'impossibilité de commander le servomoteur avec le matériel actuel :

Il est impossible de commander un servomoteur avec le matériel actuel car il ne répond pas assez vite.

Les cartes utiles n'étant pas disponibles est coûteuse, la solution technique proposée est de se servir d'une sortie analogique de l'automate et de générer le signal de commande impulsif à l'aide d'une carte Arduino :



Les impulsions générées par la carte Arduino doivent être inversées par rapport aux impulsions appliquées au moteur (présence transistor en mode commun)

Vérifier avant de mettre sous tension .

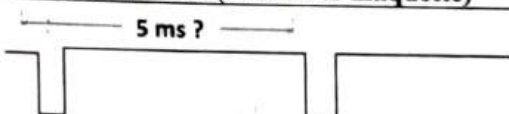
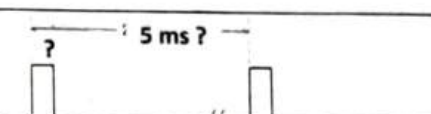
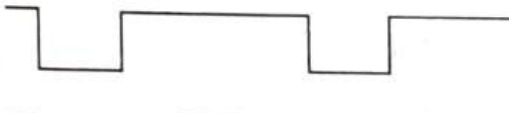
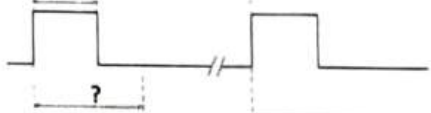

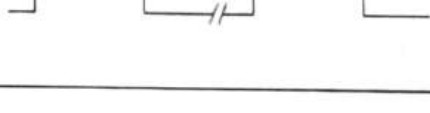
Avant de mettre sous tension nous avons vérifié à l'ohmmètre que le câblage fonctionne correctement ensuite nous avons demandé au professeur de valider notre câblage et notre méthode de vérification .

3. Programmation et envoie de la commande impulsionnelle , avec consigne manuelle

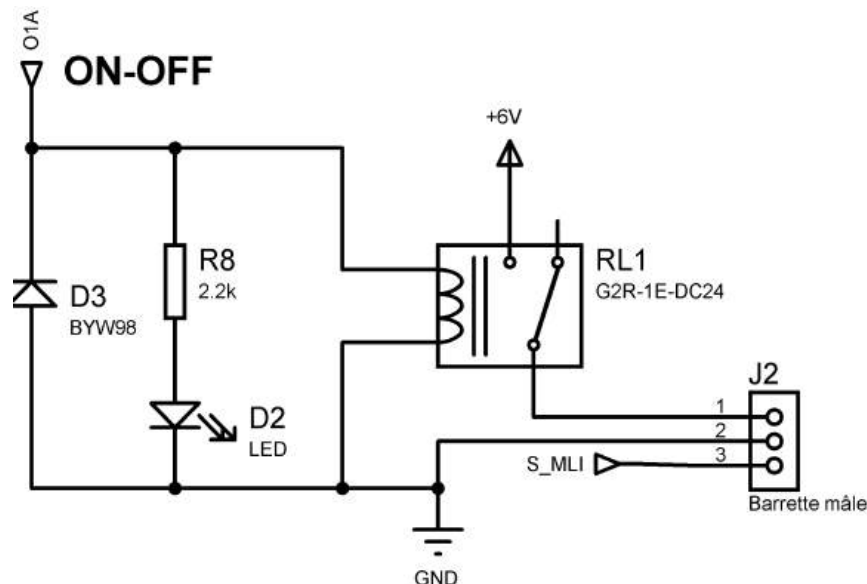
Le bornier B4 de l'interface arduino est censé recevoir le signal analogique venant de l'automate en un signal impulsionnel de commande du moteur en vitesse .

L'objectif de la carte est de transformer un signal analogique venant de l'automate en un signal impulsionnel de commande du moteur en vitesse

Cependant dans un 1er temps , pour les tests et comprendre le fonctionnement de la MLI nous allons utiliser les potentiomètres central pour commander la vitesse du moteur .

Conversion à réaliser par programmation			
Signal analogique Entrée Arduino	Signal de commande inversé Sortie Arduino (vers carte maquette)	Tension appliquée au servomoteur	Vitesse obtenue
Min = 0 V			Vitesse négative maximale
2,5 V			Vitesse nulle
Max = 5V			Vitesse positive maximale

La condition pour que le servo moteur puisse tourner est d'activer le relais via l'automate .



- Signal analogique : entrée du potentiomètre
- Période du signal : 5ms (modifiable en cas de problème)
- Largeur d'impulsion variant dans un 1er temps entre 0,5ms et 2ms

```
#define MOTEUR 2
int dure=000;

void setup() { // declaration de la vitesse et des pins E/S dans cette boucle
Serial.begin(9600); // vitesse définie a 9600 liaison série
pinMode(MOTEUR, OUTPUT); // définit la sortie de la pin moteur
pinMode(10, OUTPUT); // led définie en sortie pour test la MLI
}

void loop() {
int potentiometre=analogRead(A2); // lecture du potentiometre
int MLI=map(potentiometre, 0, 1023, 500, 2000); //map(value, fromLow, fromHigh,
toLow, toHigh); //1434 le zéro et 1300 1500
Serial.println(MLI); // lecture de notre mli
digitalWrite(MOTEUR,0); // Moteur est mis a 0
delayMicroseconds(MLI); //delay proportionnelle a la valeur de MLI
}
```



```
digitalWrite(MOTEUR,1);// Moteur est mis a 1
delayMicroseconds(5000-MLI);//delays de 5000-MLI uS
}
```

Élément à intégrer dans le programme :

- Lecture du signal d'entrée analogique variant entre 0 et 5V.
- Calcul de la largeur d'impulsion correspondante
- Envoie du signal inversé tous les 5ms

Voici le code il est affiné de manière à ce que le 0V et le 5V du signal analogique correspondent pile au moment où les vitesses maximales négatives et positives sont atteintes.

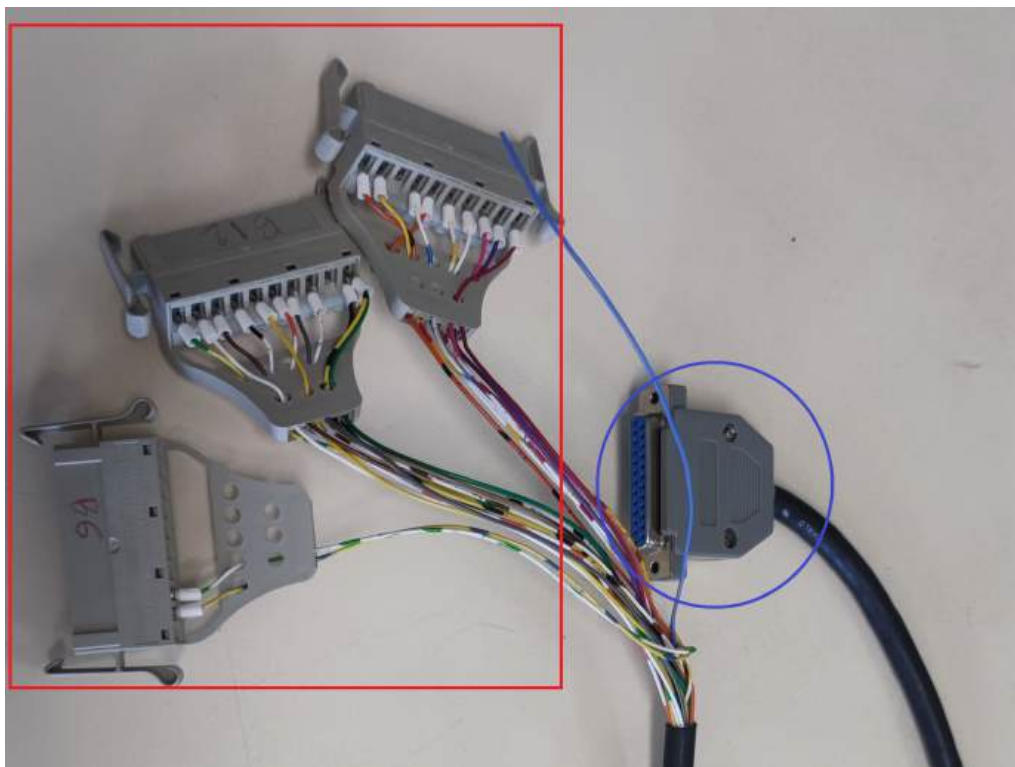
```
#define MOTEUR 2
int dure=000;
void setup(){// declaration de la vitesse et des pins E/S dans cette boucle
Serial.begin(9600); // vitesse définie a 9600 liaison série
pinMode(MOTEUR, OUTPUT);// definie la sortie de la pin moteur
pinMode(10, OUTPUT);// led definie en sortie pour test la MLI
}

void loop() {
int potentiometre=analogRead(A2);// lecture du potentiometre
int MLI=map(potentiometre, 0, 1023, 1300, 1500); //map(value, fromLow,
fromHigh, toLow, toHigh); //1434 le zéro et 1300 1500
Serial.println(MLI);// lecture de notre mli
digitalWrite(MOTEUR,0);// Moteur est mis a 0
delayMicroseconds(MLI);//delay proportionnelle a la valeur de MLI
digitalWrite(MOTEUR,1);// Moteur est mis a 1
delayMicroseconds(5000-MLI);//delays de 5000-MLI uS
}
```

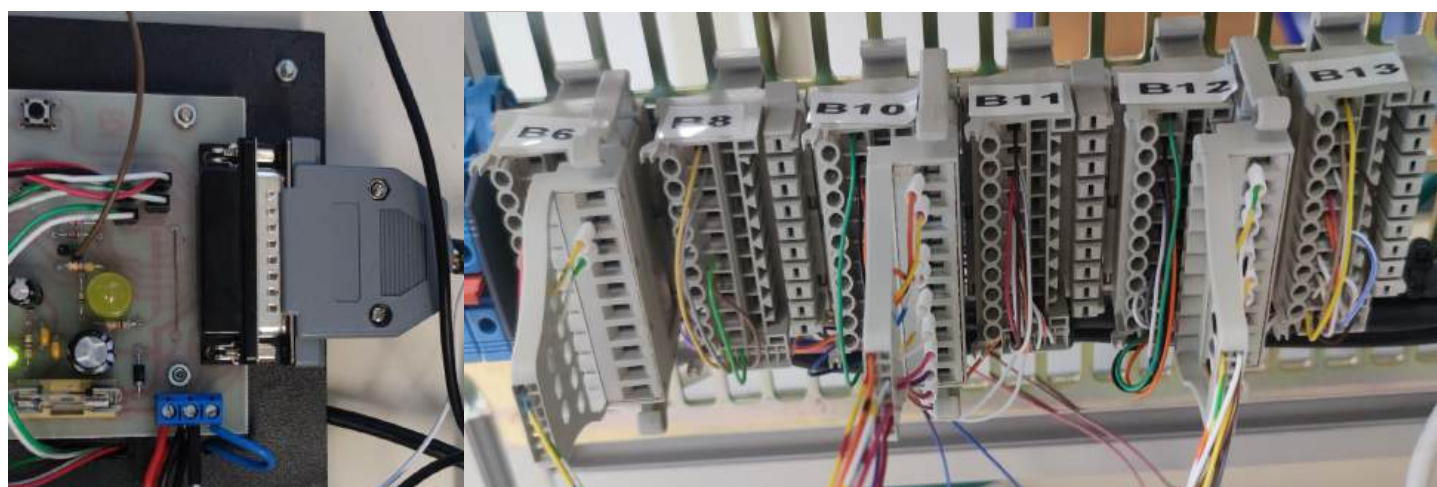
Programmation M340

1. Réalisation du raccordement entre l'automate (M340) et la maquette

Pour le raccordement il suffit de relier le câble entouré en bleu à la maquette et le reste entouré en rouge et de le brancher à l'automate en suivant l'ordre marqué de B6 à B13.



Ensuite on le branche à la maquette et à l'automate :



2. Raccordement entre M340 et l'interface Arduino

Il faut s'assurer de la compatibilité des signaux entre l'automate et la carte Arduino

La référence de la carte analogique Schneider

La référence de la carte analogique de Schneider est la "BMXAMM0600"

La tensions maximale envoyée par celle-ci

La tensions maximale envoyée par celle-ci est de 10V

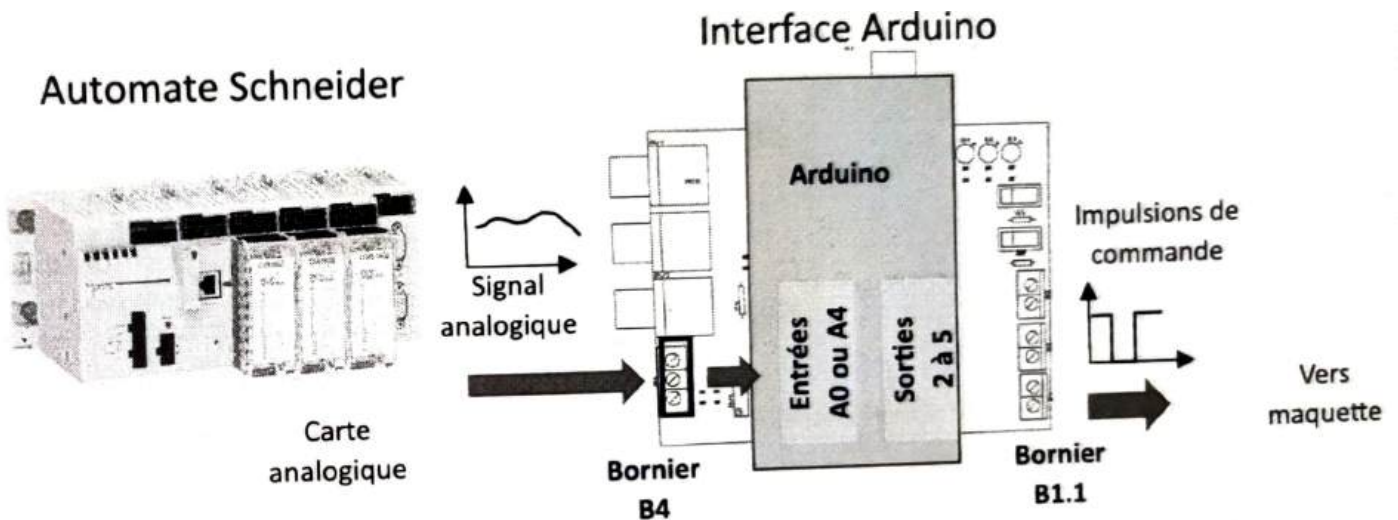
La tension maximale acceptée par les entrées analogique Arduino

La tension maximale acceptée est de 5 volts pour les entrées analogique

Conclure sur la liaison Carte analogique/Arduino

Cela reste possible si on fait une adaptation de la tension 10V à 5V sur la carte arduino ou sinon que nous dépassons pas la limite lorsque nous envoyons le signal analogique .

Le signal analogique doit être envoyé sur le bornier B4 et traité avant d'arriver sur les entrées A0 ou A4 de l'arduino .



Si le signal analogique est envoyé sur la borne B4.A0 d'après le schéma de l'interface , en négligeant la résistance R6, indiquer :

À quelle condition sur la tension d'entrée A0 de l'Arduino la diode D4 devient passante .

Quand il a 0V ou la masse.

À quelle condition sur la tension d'entrée A0 de l'Arduino la diode D4 devient passante .

Quand il a 5V ou VCC

Que se passe-t-il alors ?

Si les conditions ne sont pas réunies, les diodes bloques.

Si le signal analogique est envoyé sur la borne B4.A4 , il est transformé par le montage associant R8 et RV4 avant d'arriver sur l'entrée A4 de l'Arduino . Donner :

Le nom typique donné à ce montage associant R8 et RV4

Le montage est souvent appelé un pont diviseur de tensions.

L'expression de la tension reçue en A4 par l'Arduino en fonction de celle arrivant en B4.A4 en déduire l'utilité de ce montage.

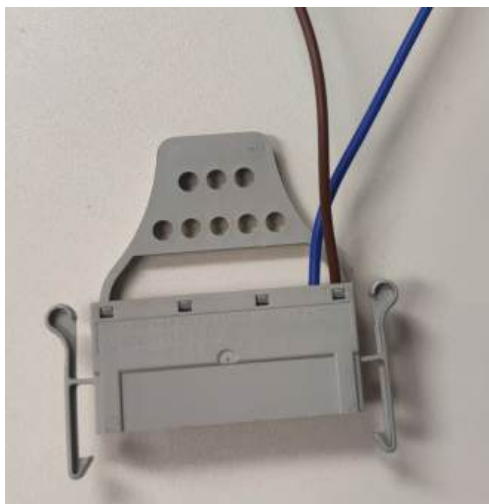
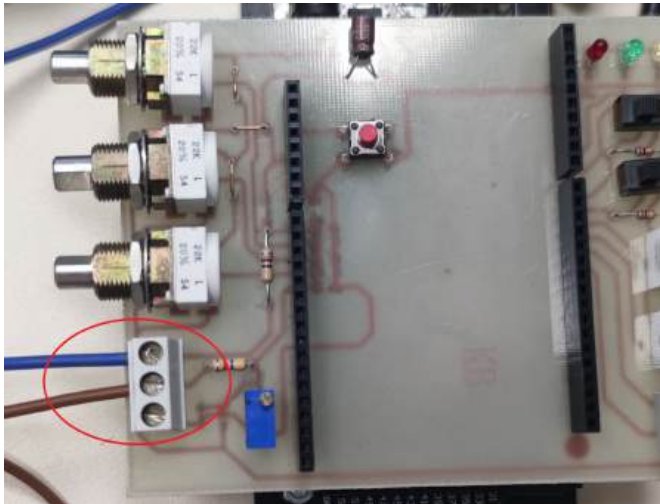
$$A4 = B4.A4 * \frac{RV4}{RV4+R8}$$
 L'utilité de ce montage est d'adapter les tensions .

Un cordon et des connecteurs sont mis à disposition.

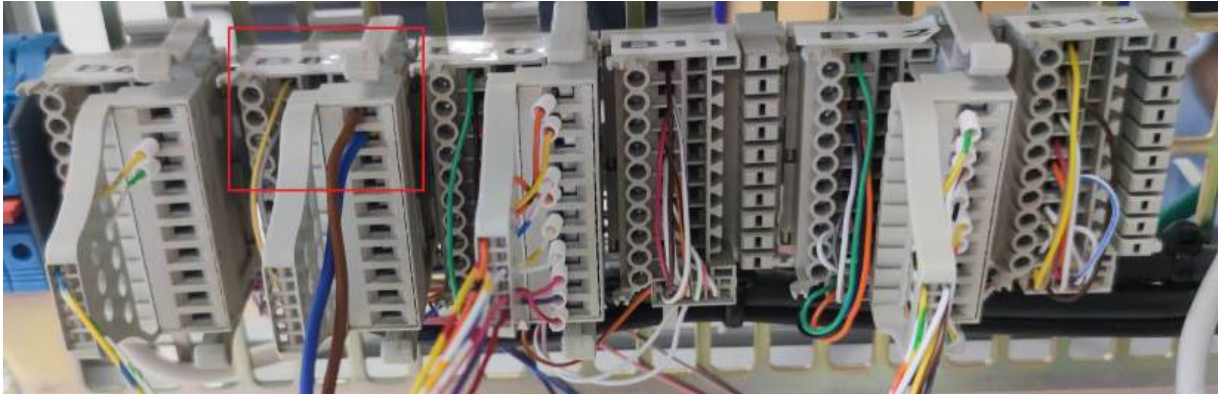
-Réaliser les branchements des entrées et des sorties TOR , en respectant les schémas

-Réaliser le branchement direct entre la carte de sortie analogique et l'entrée de l'interface Arduino(BORNE B4-A4)

Il faut brancher les entrée analogique sur le bornier B4 entouré en rouge de l'arduino puis après le reliée à l'automate.



Puis après il faut le relier à l'automate.



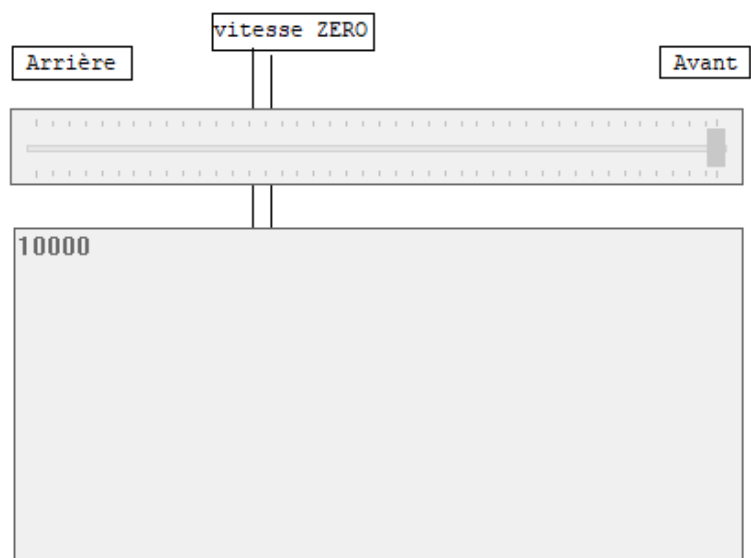
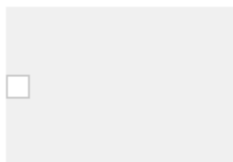
Faire vérifier

Nous avons d'abord testé à l'ohmmètre puis après fait valider par le professeur.

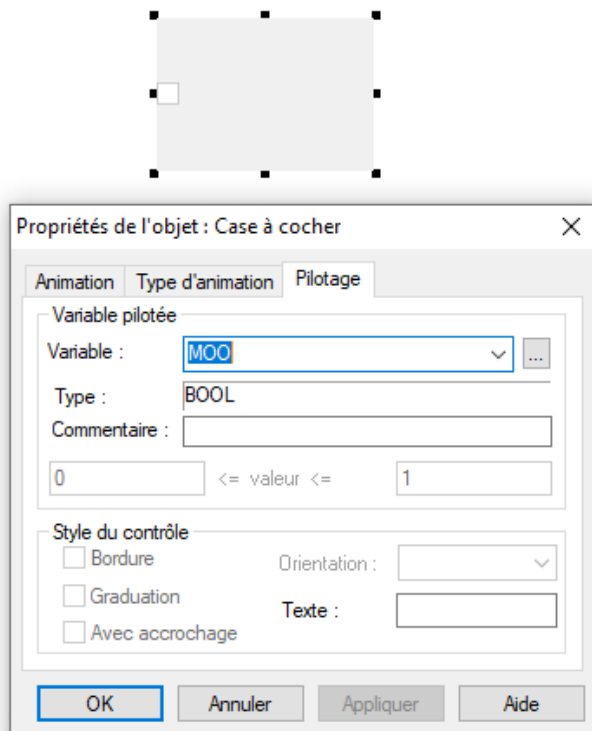
3. Vérification du fonctionnement du servomoteur depuis la commande analogique automate.

Pour vérifier le bon fonctionnement nous avons créé un écran d'exploitations pour le curseur qui aura comme rôle de gérer la vitesse et la direction du moteur et un contact pour commander la sortie MO_ON pour activer le fonctionnement du moteur (Sortie pour commande le relais).

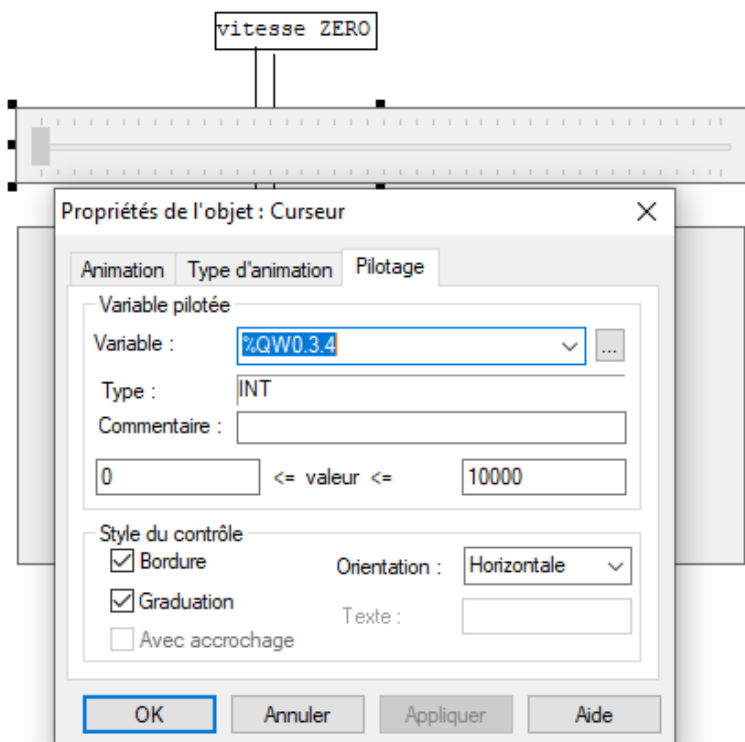
Voici l'écran de simulation a gauche le contact pour activer le moteur et à droite le curseur pour gérer la vitesse.



Ensuite voici la configuration pour que cela fonctionne correctement :
Voici les paramètres pour le contact qui activerait MO_ON .



Ensuite le paramètres du curseur qui va de 0 à 10V :



Ensuite nous avons adapté le code arduino :

```
#define MOTEUR 4

void setup() { // declaration de la vitesse et des pins E/S dans
cette boucle
Serial.begin(9600); // vitesse définie a 9600 liaison série
pinMode(MOTEUR, OUTPUT); // definie la sortie de la pin moteur
pinMode(10, OUTPUT); // led definie en sortie pour test la MLI
}

void loop() {
Serial.print("A4 :");
int potentiometre=analogRead(A4); // lecture du potentiometre
Serial.println(potentiometre);
int MLI=map(potentiometre,0,1023,1300,1500);
digitalWrite(MOTEUR,0);
delayMicroseconds(MLI);
digitalWrite(MOTEUR,1);
delayMicroseconds(5000-MLI);
}
```

Puis nous avons testé et cela fonctionne correctement .

4. Programmation en langage Grafcet

Programmer le Grafcet de façon à satisfaire le plus possible au cahier des charges, en donnant en plus la possibilité de choisir la vitesse de déplacement de la porte .

Voici les rôles de chaque entrée et sorties :

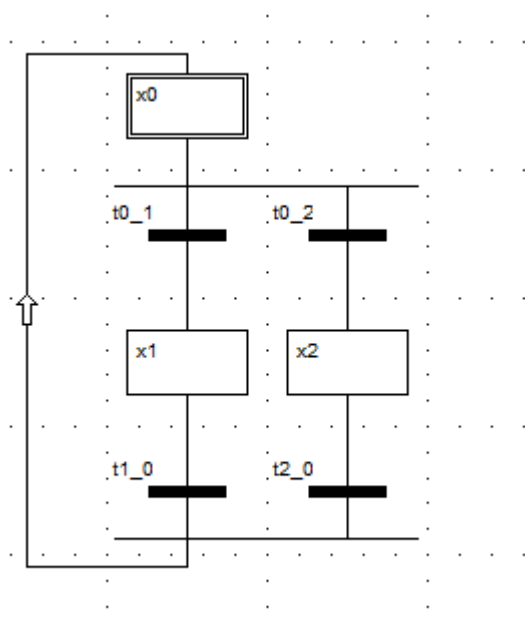
Entrée/Sorties	ACTIF	Rôle
Mo	1	BP pour monter la porte
DE	1	BP pour descendre la porte
CH	1	Capteur de fin de course haut
CB	1	Capteur de fin de course Bas

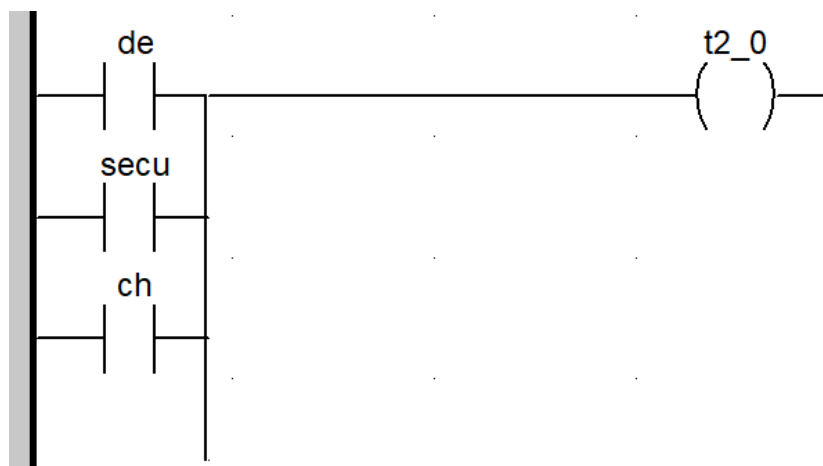
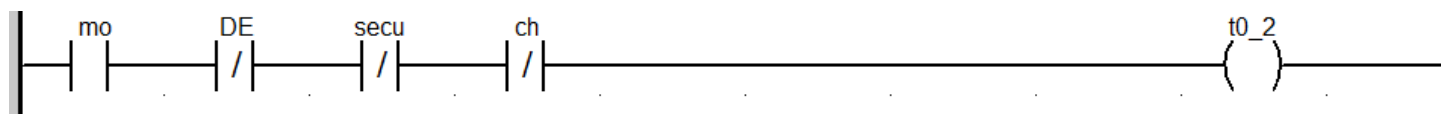
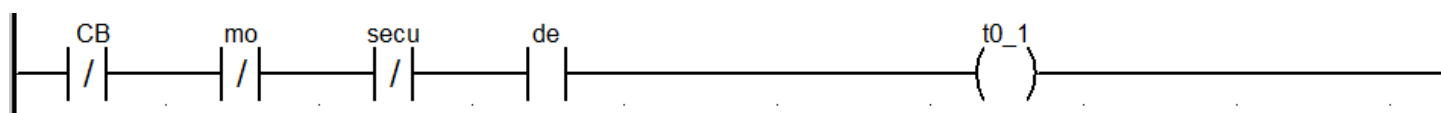
SECU	1	détecte les obstacles devant la porte du garage
------	---	---

MO_ON	1	à 1L Autorise le moteur à fonctionner
SENS_MD	1 et 0	1L pour la montée de la porte et 0L pour la descente de la porte du garage Quand moteur pas actif sortie forcément à 0L
VOYANT	1	La signalisation lumineuse.
BUZ_P	1	Signalisation sonore
BUZ_C	1	avertisseur sonore signal continu

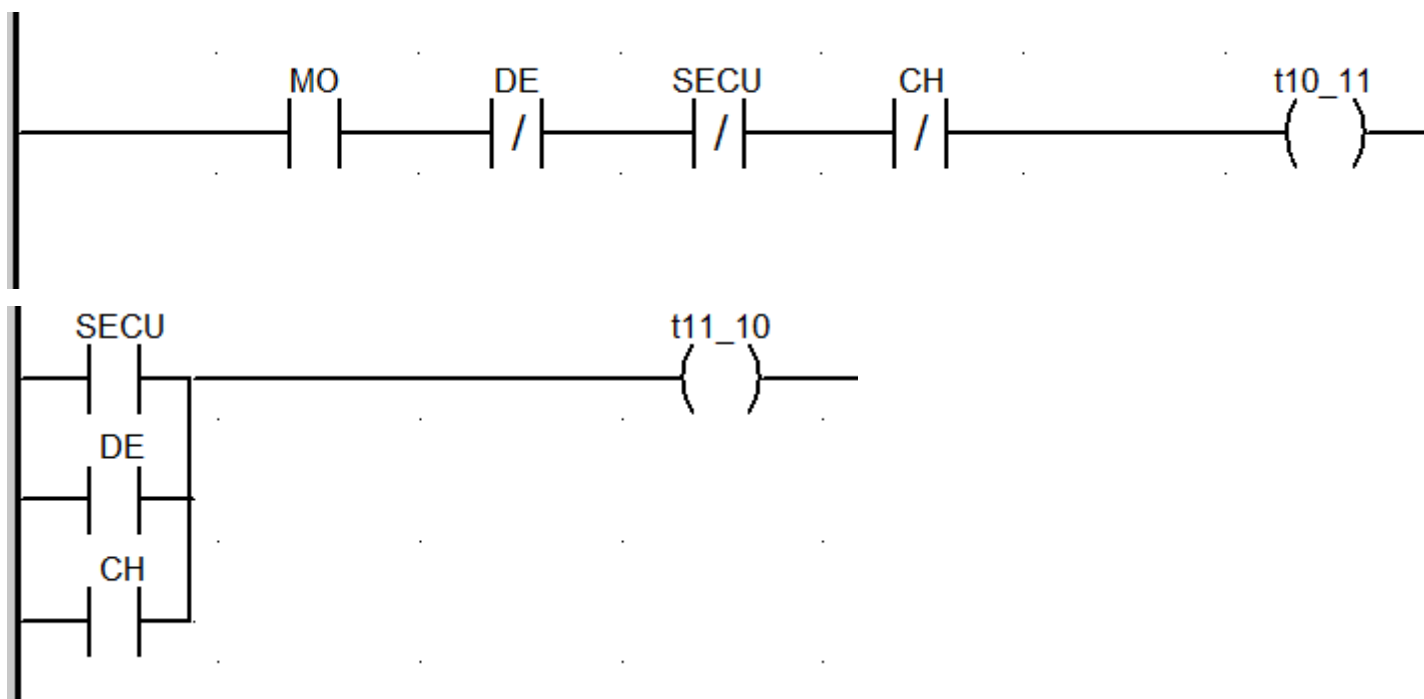
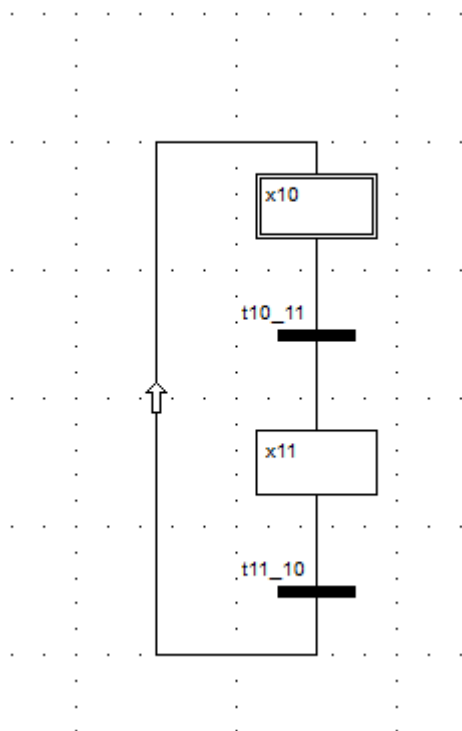
Nom	Type	Adresse	Valeur	Commentaire	Horodatage	Droits lecture/écriture (R/W) de la variable référencée	Ech
● BUZ_C	EBOOL	%Q0.2.4			Aucun		
● BUZ_P	EBOOL	%Q0.2.3			Aucun		
● CB	EBOOL	%I0.1.3			Aucun		
● CH	EBOOL	%I0.1.4			Aucun		
● CODE	EBOOL	%I0.1.7			Aucun		
● DE	EBOOL	%I0.1.1			Aucun		
● DE_E	EBOOL	%I0.1.6			Aucun		
● MO	EBOOL	%I0.1.0			Aucun		
● MO_E	EBOOL	%I0.1.5			Aucun		
● MO_ON	EBOOL	%Q0.2.1			Aucun		
● MOO	EBOOL				Aucun		
● SECU	EBOOL	%I0.1.2			Aucun		
● SENS_MD	EBOOL	%Q0.2.2			Aucun		
● IO_1	BOOL				Aucun		
● IO_2	BOOL				Aucun		
● I1_0	BOOL				Aucun		
● I2_0	BOOL				Aucun		
● I10_11	BOOL				Aucun		
● I11_10	BOOL				Aucun		
● I20_21	BOOL				Aucun		
● I20_22	BOOL				Aucun		
● I20_23	BOOL				Aucun		
● I21_20	BOOL				Aucun		
● I22_20	BOOL				Aucun		
● I23_20	BOOL				Aucun		
● I30_31	BOOL				Aucun		
● I30_32	BOOL				Aucun		
● I30_33	BOOL				Aucun		
● I31_30	BOOL				Aucun		
● I32_30	BOOL				Aucun		
● I33_30	BOOL				Aucun		
● voyant	EBOOL	%Q0.2.0			Aucun		

MO_ON et VOYANT :

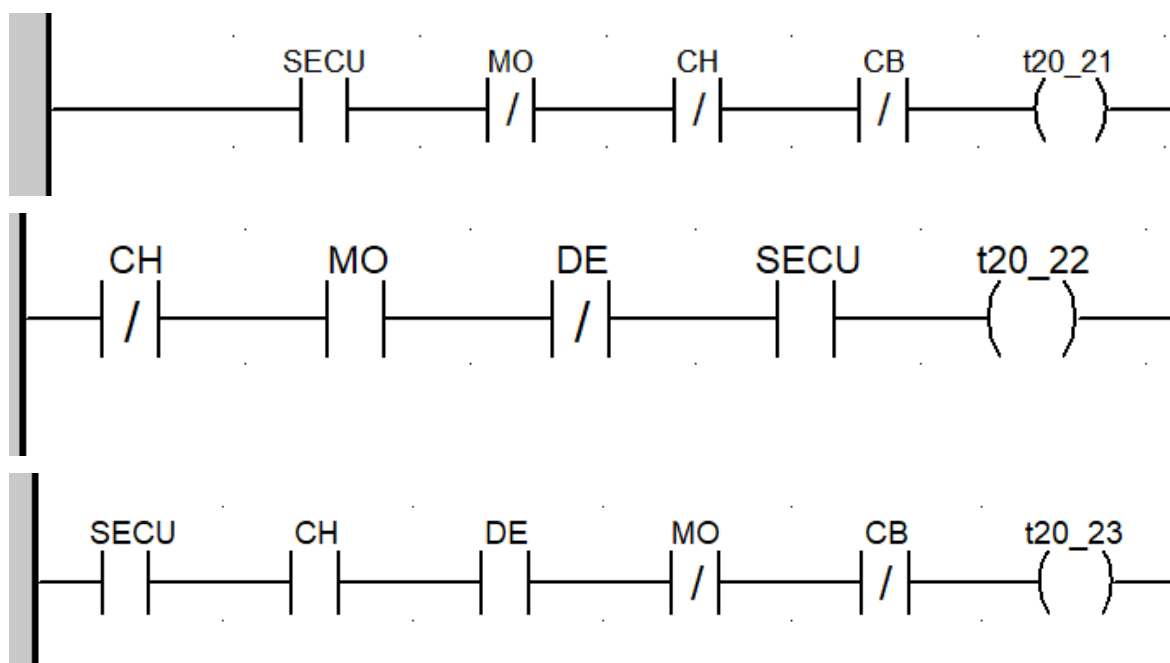
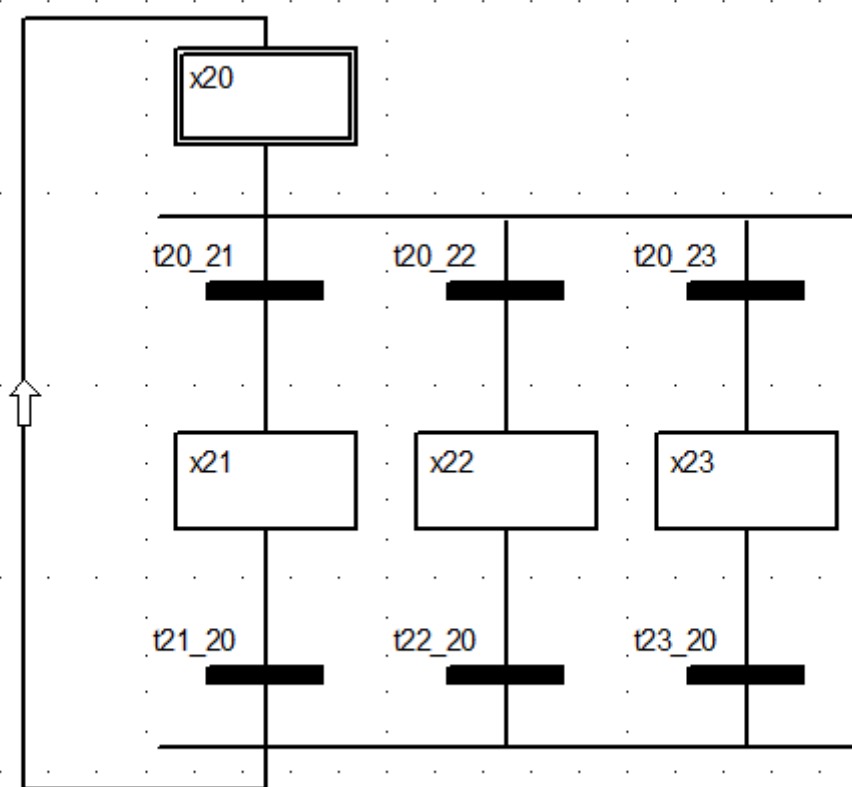


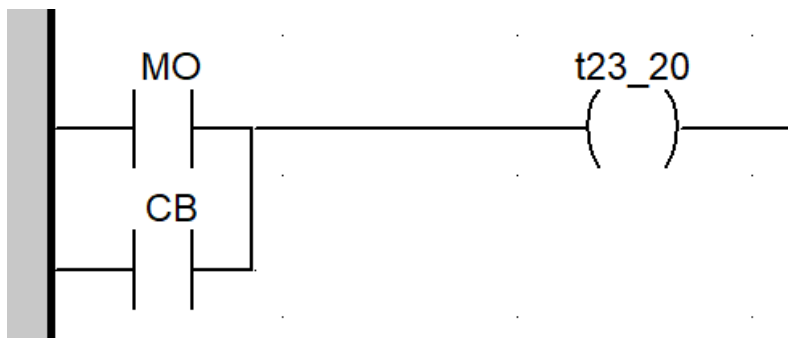
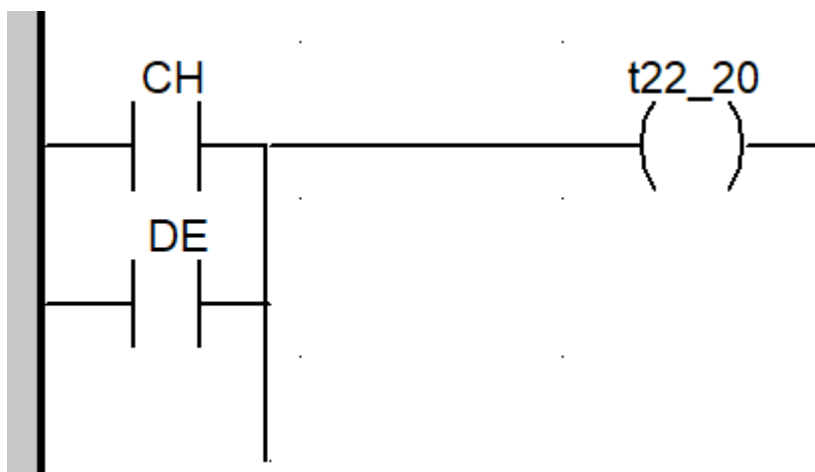
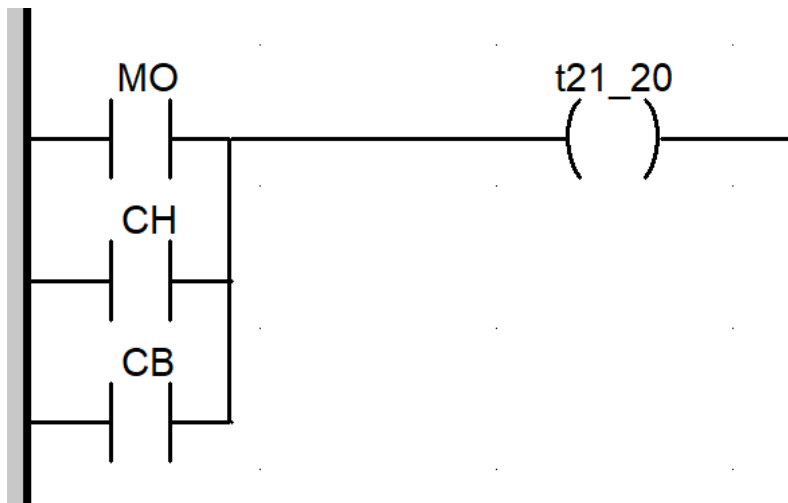


SENS_MD

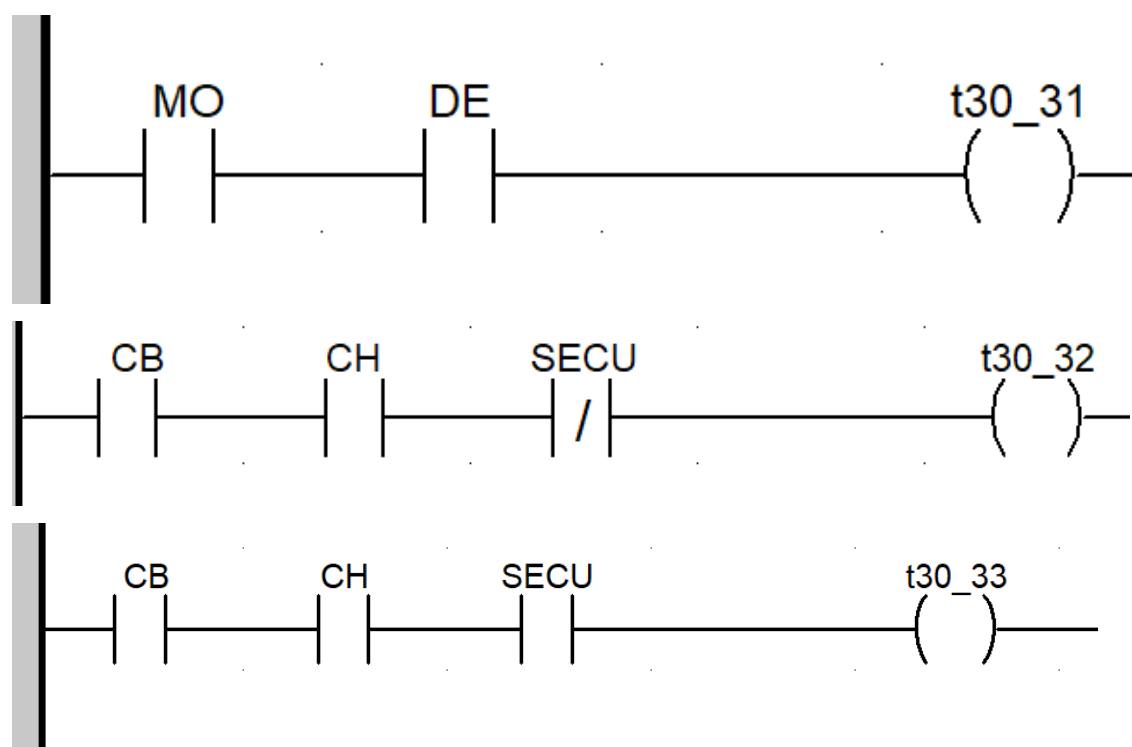
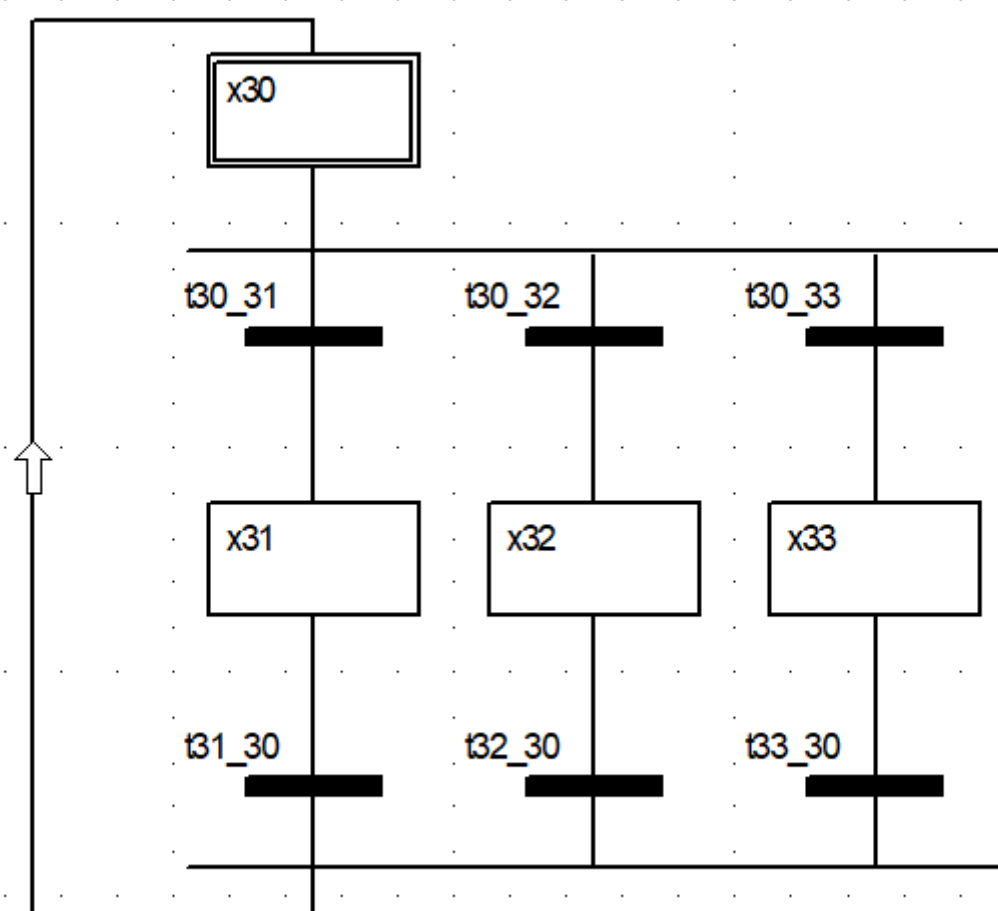


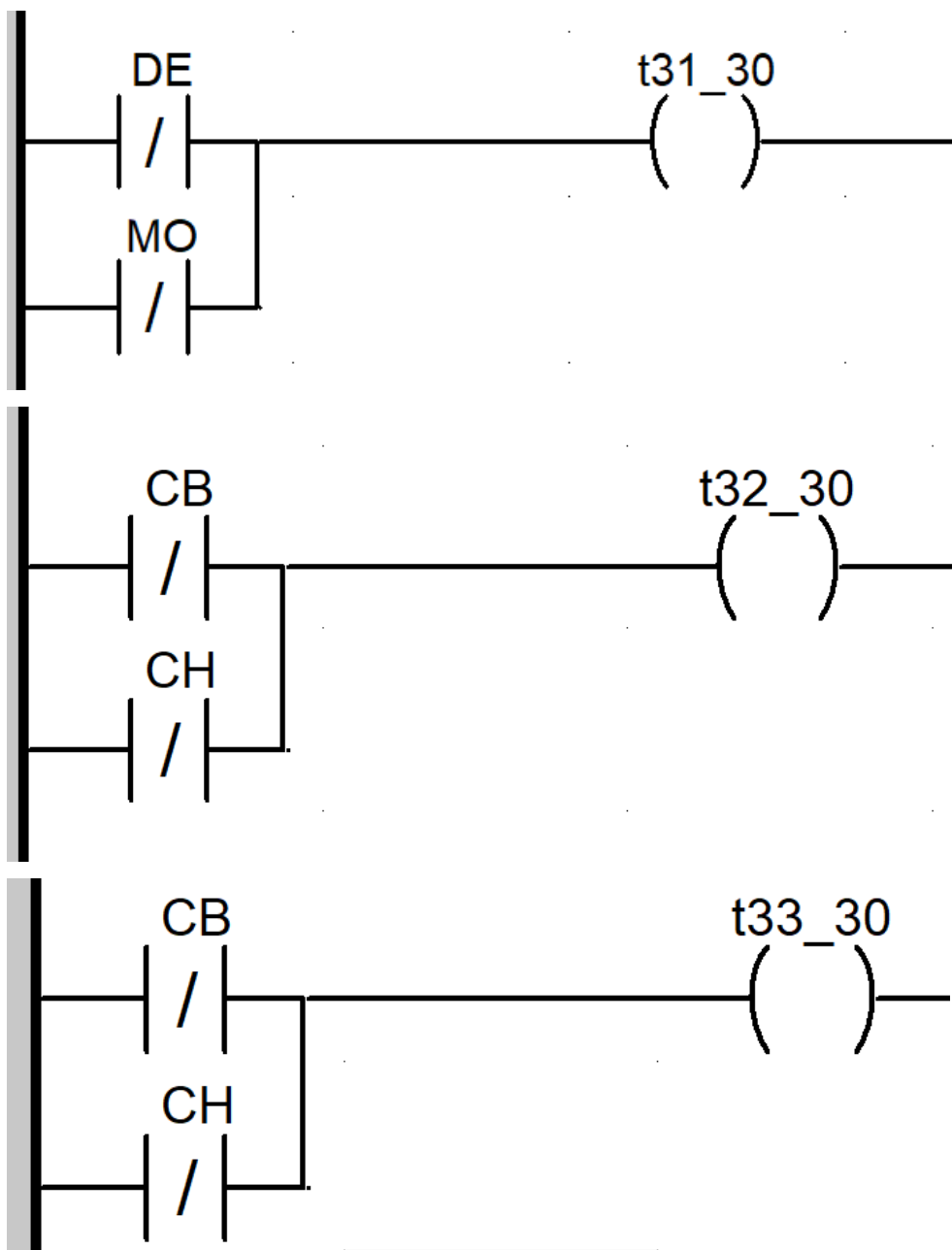
BUZ P :



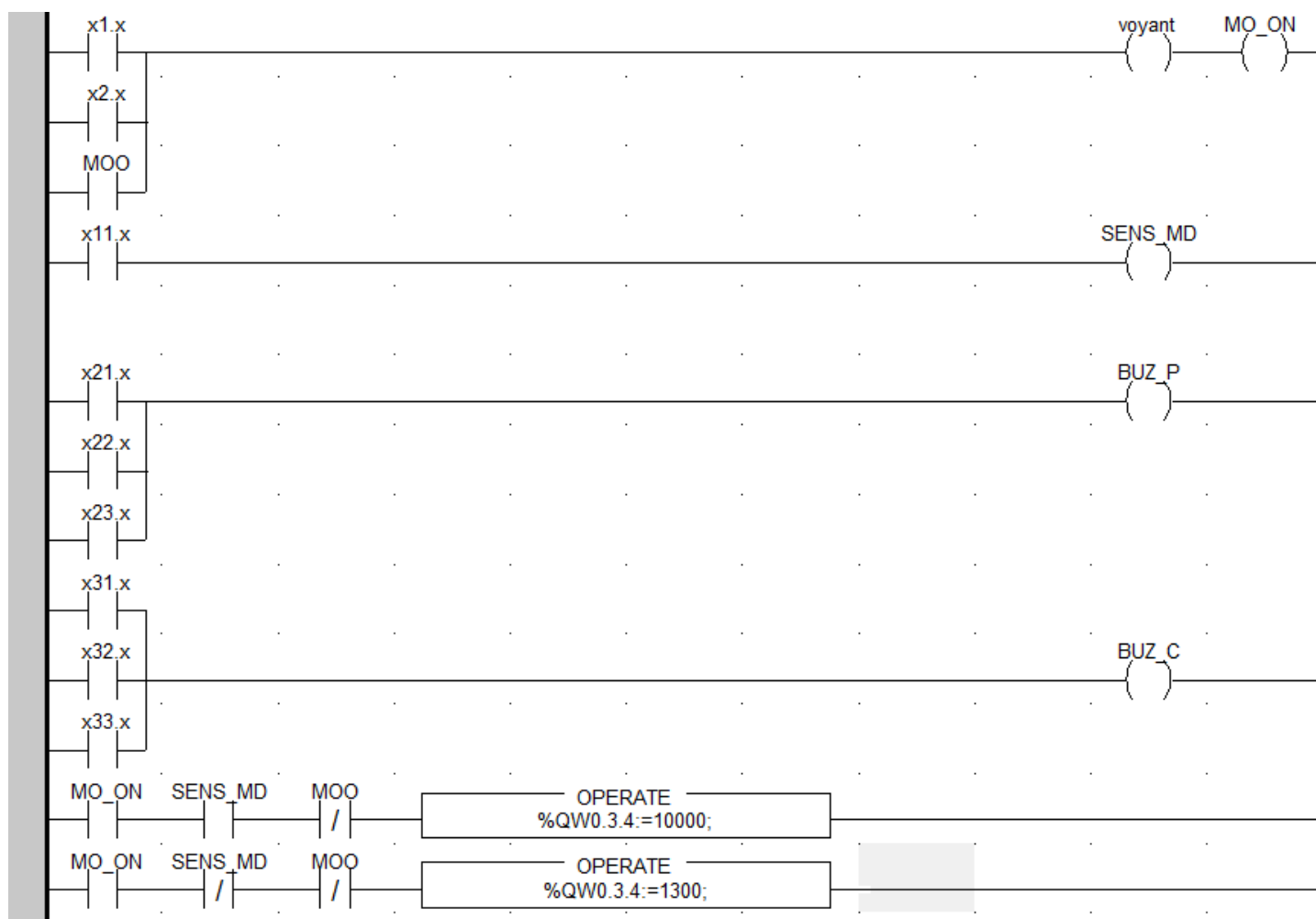


BUZ_C:



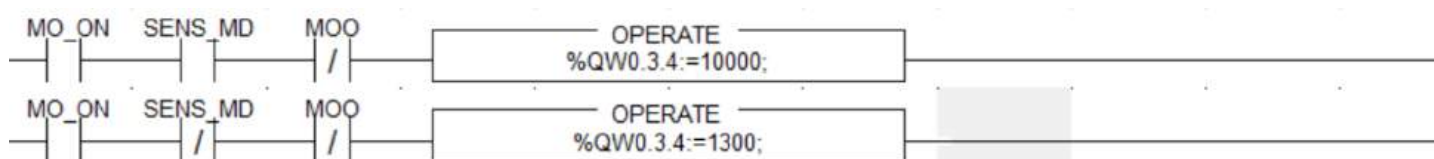


Sortie :



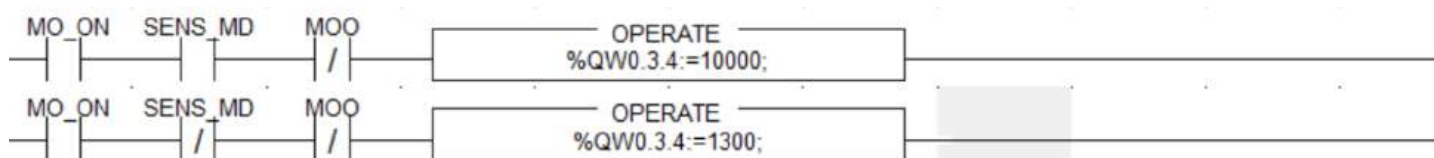
Voies d'amélioration

1. Réglage manuel des commandes limites de vitesse



On règle ici les commandes de limites de vitesse

2. Prise en compte dans le programme automate



Si le contact MOO est actif alors la gestion de la direction et de la vitesse se fait par rapport au curseur sinon il se fait manuellement par la saisie de la valeur pour la sortie analogique. Le contact MOO est expliqué à la page (12) du compte rendu.

Conclusion:

Dans cette SAE nous avons pu découvrir comment utiliser la carte arduino pour gérer le moteur de la porte dans un premier temps avec un potentiomètre pour comprendre comment fonctionne la MLI et ensuite avec un signal analogique venant de l'automate .Le SAE a été utile pour nous apprendre à communiquer entre l'automate et d'autres appareillages mais aussi à continuer notre formation sur le GRAFCET.